TREE-BASED CERTIFICATE REVOCATION SYSTEM

INS AAI

This application is based on U.S. provisional patent application No. 60/006,143 filed on November 2, 1995.

TECHNICAL FIELD

The present invention relates generally to secure communications and more particularly to schemes for certificate management.

BACKGROUND OF THE INVENTION

In many settings, it is necessary to certify certain data, as well as to revoke already issued certificates. For instance, in a Public-Key Infrastructure, (PKI) it is necessary to certify users' public keys.

In a digital signature scheme, each user U chooses a signing key SK_u and a matching verification key, PK_u. User U uses SK_u to compute easily his digital signature of a message m, $SIG_u(m)$, while anyone knowing that PK_u is U's public key can verify that $SIG_u(m)$ is U's signature of m. Finding $SIG_u(m)$ without knowing SK_u is practically impossible. On the other hand, knowledge of PK_u does not give any practical advantage in computing SK_u. For this reason, it is in U's interest to keep SK_u secret (so that only he can digitally sign for U) and to make PK_u as public as possible (so that everyone dealing with U can verify U's digital signatures). At the same time, in a world with millions of users, it is essential in the smooth flow of business and communications to be certain that PK_u really is the legitimate key of

user U. To this end, users' public keys are "certified." At the same time it is also necessary to revoke some of the already-issued certificates.

CERTIFICATION AND REVOCATION OF PUBLIC KEYS. Typically, certificates for users' public keys are produced and revoked by certifying authorities called CA's. A complete public-key infrastructure may involve other authorities (e.g., PCAs) who may also provide similar services (e.g., they may certify the public keys of their CA's). The present inventions can be easily applied to such other authorities. A CA can be considered to be a trusted agent having an already certified (or universally known) public key. To certify that PK_u is U's public key, a CA typically digitally signs PK_u together with (e.g., concatenating it with) U's name, a certificate serial number, the current date (i.e., the certification or issue date), and an expiration date. (Before certifying U's public key, it is necessary to perform additional steps, such as properly identifying user U. The present invention, however, does not depend on these additional steps). The CA's signature of PK_u is then inserted in a Directory and/or given to U himself.

Upon receiving the (alleged) digital signature of user U of a message M, $SIG_u(M)$, a recipient R needs to obtain a certificate for PK_u. (Indeed, $SIG_u(M)$ may be a correct digital signature of M with respect to some public key PK_u, but R has no guarantee that PK_u is indeed U's public key). Recipient R may obtain this certificate from the Directory, or from his own memory (if he has previously cached it), or from U himself. Having done this, R verifies (1) the correctness of the CA's certificate for PK_u with respect to the CA's public

key, and (2) the correctness of $SIG_u(M)$ with respect to PK_u. (If the CA's public key is not universally known, or cached with R, then a certificate for this key too must be obtained.)

Certificate retrieval is thus possible, although not necessarily cheap. Unfortunately, however, this is not the only retrieval that R needs to do. Indeed, it is crucially important that R makes sure that the certificate for PK_u has not been revoked. This check, of course, may not be needed after the certificate's expiration date, but is needed during the certificate's alleged lifetime. A user's certificate can be revoked for a variety of reasons, including key compromise and the fact that the user is no longer associated with a particular CA.

To enable a recipient to establish whether a given certificate has been revoked, it is known that each CA periodically issues and gives the Directory a Certificate Revocation List (CRL for short), in general containing an indication of all the (not yet expired) certificates originally issued by it. A CRL typically consists of the issuer's digital signature of (1) a *header* comprising the issuer's name (as well as the type of his signature algorithm), the current date, the date of the last update, and the date of the next update, together with (2) a *complete* list of revoked certificates (whose date has not yet expired), each with its serial number and revocation date. Since it is expected that a CA revokes many of her certificates, a CRL is expected to be quite long.

After performing some checks on the CA's CRL (e.g., checking the CA's digital signature, checking that the CRL has arrived at the expected time, that a certificate declared

revoked in the previous CRL of that CA - and not yet expired - still is revoked in the current CRL, etc.), the Directory stores it under its CA name.

When a user queries the Directory about the revocation of a certificate issued by a given CA, the Directory responds by sending to the user the latest CRL of that CA. The user can then check the CRL signature, the CRL dates (so as to receive a reasonable assurance that he is dealing with the latest one), and whether or not the certificate of interest to him belongs to it.

While CRLs are quite effective in helping users establishing which certificates are no longer deemed valid, they are also extremely expensive, because they tend to be very long and need to be transmitted very often.

The National Institute of Standards and Technology has tasked the MITRE Corporation to study the organization and cost of a PKI for the Federal Government. This study estimates that CRLs constitute by far the largest entry in the Federal PKI's cost list. According to MITRE's estimates/assumptions, in the Federal PKI there are about three million users, each CA serves 30,000 users, 10% of the certificates are revoked (5% because of key compromise and 5% because of change in affiliation with the organization connected to a given CA). CRLs are sent out bi-weekly, and, finally, the recipient of a digital signature requests certificate information 20% of the time. (The remaining 80% of the time he will be dealing with public keys in his cache). The study envisages that each revoked certificate is specified in a CRL by means of about 9 bytes: 20 bits of serial number and 48

bits of revocation date. Thus, in the Federal PKI, each CRL is expected to comprise thousands of certificate serial numbers and their revocation dates; the header, however, has a fixed length, consisting of just 51 bytes.

At 2 cents per kilobyte, the impact of CRL transmission on the estimated yearly costs of running the Federal PKI is stunning: if each federal employee verifies 100 digital signatures per day on average, then the total PKI yearly costs are $10,848 Millions, of which $10,237 Millions are due to CRL transmission. If each employee is assumed to verify just 5 digital signatures a day on average, then the total PKI yearly costs are $732 Millions, of which 563 Millions are due to CRL transmission.

The MITRE study thus suggests that any effort should be made to find alternative and cheaper CRL designs. This is indeed the goal of the present invention.

BRIEF SUMMARY OF THE INVENTION

To avoid the dramatic CRL costs, a novel *Certification Revocation System* is described, where requesting users no longer receive the latest list of revoked certificates (of a given CA). The scheme utilizes a known tree-based authentication technique in a novel way to overcome the problems associated with the prior art.

It is thus a primary object of this invention to provide certificate management without providing CRL's to a user requesting information about the certificate (e.g, its validity).

Although special CRL's still may be used between CA's and the Directory in this scheme, the tree-based technique allows the Directory to convince users of whether or not a given certificate is still valid in a way that is essentially individualized, and thus quite convenient.

DETAILED DESCRIPTION AND PREFERRED EMBODIMENT

A known scheme for authenticating an item in a list of items is described in U.S. Patent No. 4,309,569 to Merkle, and the disclosure therein is hereby incorporated by reference. Familiarity with the Merkle scheme is assumed in the following discussion.

By way of brief background, assume that the set S of certificates consists of n items, where, for convenience, n is a power to two: $n = 2^k$ (else, we may introduce "dummy" items so that this is the case). The set organizer sorts (e.g., lexicographically) the items in S and then tree-hashes them as taught by Merkle. That is, let H be a one-way hash function, mapping strings of any length into B-bit strings (e.g., B = 200). Then conceptually, the organizer stores the ith item, I_i, into the ith leaf of a full binary tree with n leaves. Then, he fills the remaining nodes of the tree in a bottom-up fashion, by storing in every internal node the hash of the concatenation of the content of its two children. In order to develop a minimum of terminology, let N be an internal node where the left child is L and the right child is R, and let V_L be the value stored in L and V_R the value stored in R (we shall refer to V_L and V_R as, respectively, a left-value and a right-value). Then, the organizer stores in node N the B-bit value $H(V_L V_R)$.

In one embodiment of the invention it is assumed that there are two (2) distinct trees (or more precisely, "tree-hashes"), a first tree in which information about issued but non-revoked certificates is stored, and a second tree in which information about revoked certificates is stored. Thus, for example, if the system has had 30,000 certificates issued but 3,000 of these have been revoked, the first tree would include information about the 27,000 certificates that remain valid while the second tree would store the information about the 3,000 revoked certificates. Thus, in this embodiment there is a tree-hash for the non-revoked certificates and a separate tree-hash for the revoked ones. In such case, the root values of these trees may be digitally signed by a CA (together with other information deemed proper) either separately or together; and the Directory uses the first tree to prove to a user that a given certificate is still deemed valid, and the second tree to prove that a certificate has been revoked. The CA builds both trees so that a given certificate (serial number) does not belong to both, but the Directory can also check for this property. Again, this and other operations may be facilitated if the serial number or any representation of the certificate that is deemed proper are ordered when inserted in the tree leaves. Among the various representations of a certificate, of course, one may use the certificate itself.

While the above technique is advantageous, in the preferred embodiment each leaf of the tree-hash stores the serial number of an issued and not-yet-expired certificate, *together* with an additional bit indicating whether or not the corresponding certificate is no longer valid (i.e., it has been revoked), and, if so, the revocation date. Of course, additional information can be stored about each or some of the serial numbers (e.g., certification date, etc.). Also, some of this information (e.g., the revocation date) can be removed or changed,

and some other way of referring to a certificate can be used rather than the serial number, although within the scope of the present invention.

Thus in this case the same tree-hash is used to represent the set of issued and not-yet-expired certificates as well as the set of revoked (not-yet-expired) certificates.

In either embodiment, a problem may arise when the user queries the Directory with a serial number not corresponding to any issued certificate. (Indeed, while many times the user has already seen a certificate and accesses the Directory just to confirm the current validity of that certificate, at other times the user wishes to obtain the corresponding certificate from the Directory). If the corresponding certificate "does not exist," the Directory is at a loss to proceed. If the Directory responds truthfully, it may not be believed by the user. If the Directory gives the users all the certificates in its possession (or those relative to a given CA) the user may suspect that the Directory left out the certificate of interest. This problem exists as well in a CRL-based system. Indeed, even if the Directory gives the user all possible tree-based information or just the latest CRL of a given CA, this does not prove to the user that the certificate in question does not exist. (In fact, the actions of the Directory may actually be interpreted as saying that the certificate is valid because it does not appear to have been revoked.) Thus in this thorny situation the Directory would have to be trusted.

To rectify this problem, it is desired to have the Directory digitally sign that the certificate specified by the user is not among the existing or the not-yet-expired ones. This

scheme at least may prevent frivolous answers by putting some liability on the Directory. Better yet, it is preferred that a CA periodically give the Directory a tree-hash specification of the status of all possible serial numbers at that time. For instance, because it is envisioned that a serial number consists of bits, the CA stores these serial numbers in the leaves of a depth-20 (full) tree-hash, together with an additional bit: 0 if not-yet-expired corresponding certificate exists, and 1 otherwise. After completing the hashing, the CA will digitally sign the root value (alone, or together with the root values of some other tree-hash). This enables the Directory to prove to the user that a serial number does not correspond to any issued and not-yet-expired certificate, namely, by giving the user the digital signature of the root value of this last tree-hash, and then the content of the leaf containing the serial number (or other specification) and the content of the siblings of the path from that leaf to the root.

Just like in the preferred embodiment of the underlying tree-hash scheme itself, one may have two separate trees for this purpose: one tree-hashing the serial numbers of issued and not-yet-expired certificates, and the other tree-hashing the serial numbers not corresponding to issued and not-yet-expired certificates. Of course, one may wish instead to merge together all possible tree-hashes mentioned so far (and possibly others) by having a CA generate a single tree-hash with 2^{20} leaves (i.e., with as many leaves as possible serial numbers), where each leaf stores the corresponding serial number, together with information specifying whether that serial number has been allocated or not, and, if so, whether its certificate has been revoked or not, and possibly other information (e.g, in case of revocation, the revocation date).

It should be appreciated that the above envisions each leaf corresponding to a serial number for simplicity only. For instance if all the information deemed appropriate about a serial number is no more than 80 bits, each leaf may deal with two serial numbers, provided of course that the information is properly coded so as not to generate confusions, errors or ambiguity. Smaller tree hashes may be built this way.

It should further be realized this type of digital signature allows batch processing. Namely, rather than signing n quantities separately, a signer tree-hashes them so as to obtain a single root-value, and then digitally signs this root-value (together with additional quantities if desired) when requested to give the signature of quantity i, the signer gives his digital signature of the root, and then the content of leaf i, and the content of the sibling of the party from that leaf to the root. This is a way to substitute n individual signatures (that can be expensive to obtain) with just one signature and n—1 hashing (which are not expensive at all). Notice that the signer does not need to be the one who supplies individualized signatures; if he transfers the n quantities (or the entire tree hash) together with his signature of the root to someone else, he may supply the signatures on demand. Notice that any signature scheme can be used for the root, including pen-written signatures, rather than digital ones.

The foregoing has outlined some of the more pertinent objects and details of a preferred embodiment of the present invention. These objects and details should be construed to be merely illustrative of some of the more prominent features and applications of the invention. Many other beneficial results can be attained by applying the disclosed

invention in a different manner or modifying the invention as will be described. Those skilled in the art will recognize that the invention can be practiced, with modification, in other and different certification methods and schemes within the spirit and scope of the invention.

One of the preferred implementations of the various routines disclosed above is as a set of instructions in a code module resident in the random access memory of a computer. Until required by the computer, the set of instructions may be stored in another computer memory, for example, in a hard disk drive, or in a removable memory such as an optical disk (for eventual use in a CD ROM) or floppy disk (for eventual use in a floppy disk drive). In addition, although the various methods described are conveniently implemented in a general purpose computer selectively activated or reconfigured by software, one of ordinary skill in the art would also recognize that such methods may be carried out in hardware, in firmware, or in more specialized apparatus constructed to perform the required method steps.

While the invention has been disclosed in connection with the preferred embodiments shown and described in detail, various modifications and improvements thereon will become readily apparent to those skilled in the art. Accordingly, the spirit and scope of the present invention is to be limited only by the following claim(s).